

AD-A278 503



DTIC  
ELECTE  
APR 25 1994

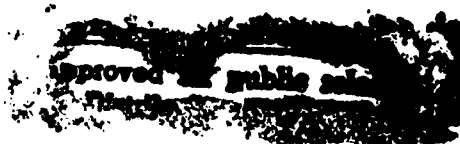
S G D

AN INTEGER SOLUTION HEURISTIC FOR  
THE ARSENAL EXCHANGE MODEL (AEM)

THESIS

Daniel Joseph Green  
Captain, USAF

AFIT/GST/ENS/94M-04



DEPARTMENT OF THE AIR FORCE  
AIR UNIVERSITY

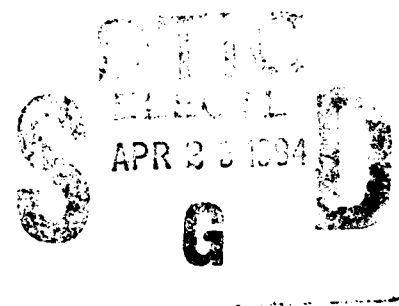
**AIR FORCE INSTITUTE OF TECHNOLOGY**

DTIC QUALITY INSPECTED 3

Wright-Patterson Air Force Base, Ohio

01 22 011

AFIT/GST/ENS/94M-04



AN INTEGER SOLUTION HEURISTIC FOR  
THE ARSENAL EXCHANGE MODEL (AEM)

THESIS  
Daniel Joseph Green  
Captain, USAF

AFIT/GST/ENS/94M-04

94-12340



5286

Approved for public release; distribution unlimited

94 4 22 011

AFIT/GST/ENS/94M-04

AN INTEGER SOLUTION HEURISTIC FOR THE ARSENAL EXCHANGE MODEL  
(AEM)

THESIS

Presented to the Faculty of the Graduate School of  
Engineering of the Air Force Institute of Technology  
Air University

In Partial Fulfillment of the  
Requirements for the Degree of  
Master of Science in Operations Research

Daniel Joseph Green, B.S.  
Captain, USAF

March, 1994

Accession For	
NTIS	<input checked="checked" type="checkbox"/>
CRA&I	<input checked="checked" type="checkbox"/>
DTIC	<input checked="checked" type="checkbox"/>
TAB	<input checked="checked" type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification .....	
By .....	
Distribution /	
Availability Codes	
Dist	Avail and/or Special
A-1	

Approved for public release; distribution unlimited

THESIS APPROVAL

STUDENT: Capt Daniel J. Green

CLASS: GST-94M

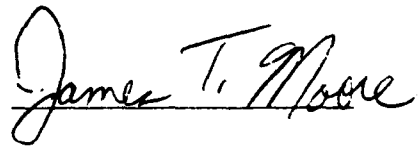
THESIS TITLE: AN INTEGER SOLUTION HEURISTIC FOR THE ARSENAL  
EXCHANGE MODEL (AEM)

DEFENSE DATE: March 2, 1994

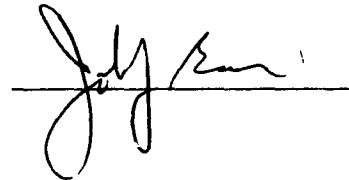
COMMITTEE: NAME/DEPARTMENT

SIGNATURE

Advisor Lt Col James T. Moore/ENS

A handwritten signature in cursive script, reading "James T. Moore", written over a horizontal line.

Reader Maj John J. Borsi/ENS

A handwritten signature in cursive script, reading "John J. Borsi", written over a horizontal line.

### *Acknowledgements*

I have had a great deal of help in writing this thesis. I would especially like to thank Mr. William Cotsworth who was always available to discuss ideas and answer any questions I had about AEM. I also owe a great deal of thanks to my advisor, Lt Col James Moore, for his help and guidance over the last six months. Thanks also to my reader, Maj John Borsi, for his insightful and timely inputs. I would also like to thank my contact at Studies and Analyses, Capt Justin Moul, for his help in seeing that I received the material support necessary to successfully complete this research. Most of all, I would like to thank my wife, Anne, for her support and encouragement.

Daniel J. Green

## Table of Contents

	Page
Acknowledgements . . . . .	ii
List of Figures . . . . .	v
List of Tables . . . . .	vi
Abstract . . . . .	vii
I. Introduction . . . . .	1
1.1 Background . . . . .	1
1.2 AEM Description . . . . .	2
1.2.1 User Inputs . . . . .	2
1.2.2 Processing . . . . .	3
1.2.3 Model Formulation . . . . .	3
1.2.4 Solution and Output . . . . .	5
1.3 Problem Definition . . . . .	5
1.4 Scope . . . . .	6
1.5 Format . . . . .	6
II. Literature Review . . . . .	7
2.1 Introduction . . . . .	7
2.2 Goal Programming . . . . .	7
2.3 The Generalized Assignment Problem . . . . .	10
2.4 Conclusion . . . . .	11
III. Goal Programming in AEM . . . . .	12
3.1 Introduction . . . . .	12
3.2 Goals . . . . .	12
3.3 Preemptive Goal Programming . . . . .	14
IV. Methodology . . . . .	17
4.1 Introduction . . . . .	17
4.2 The Optimal Integer Solution . . . . .	17
4.3 Current Solution Method . . . . .	19

	Page
4.4 A Better Rounding Method . . . . .	19
4.5 Model Formulation . . . . .	21
4.6 Solution Procedure . . . . .	23
4.7 Revised Solution Method . . . . .	27
4.8 Revised Solution Procedure . . . . .	27
V. Implementation and Testing . . . . .	31
5.1 Implementation . . . . .	31
5.2 Testing . . . . .	31
5.2.1 The First Test Case . . . . .	31
5.2.2 The Second Test Case . . . . .	34
5.2.3 The Third Test Case . . . . .	36
5.2.4 The Actual Case . . . . .	38
VI. Conclusions and Recommendations . . . . .	39
6.1 Conclusions . . . . .	39
6.2 Recommendations . . . . .	39
Bibliography . . . . .	41
Vita . . . . .	42

*List of Figures*

Figure	Page
1. Flow Diagram, Solution Procedure . . . . .	26
2. Flow Diagram, Revised Solution Procedure . . . . .	30



## *List of Tables*

Table	Page
1. Goal Performance, Test Case 1 . . . . .	32
2. Damage Expectancy and Target Coverage, Test Case 1 . . . .	33
3. Goal Performance, Test Case 2 . . . . .	35
4. Damage Expectancy and Target Coverage, Test Case 2 . . . .	36
5. Goal Performance, Test Case 3 . . . . .	37
6. Damage Expectancy and Target Coverage, Test Case 3 . . . .	38
7. Performance Summary for Actual Case . . . . .	38

*Abstract*

The Air Force Studies and Analyses Agency sought a method for converting the continuous solution produced by the Arsenal Exchange Model into a feasible integer solution. The current integerization method leaves weapons unused and targets uncovered and could be improved. In the method developed, the noninteger valued variables in the continuous solution are rounded down to the nearest integer to produce a truncated solution. An integer goal program is then used to reallocate the weapons and targets made available in the rounding process. The truncated solution is then combined with the solution of the integer goal program to produce a feasible integer solution for the original problem. The revised solution method, using the LP relaxation of the integer goal program, was implemented in AEM. The implementation was used to solve four test cases. In all four cases, the revised solution method produced solutions that were closer to the continuous solution in terms of damage expectancy, target coverage, and goal performance than the solutions produced by the current method.

# AN INTEGER SOLUTION HEURISTIC FOR THE ARSENAL EXCHANGE MODEL (AEM)

## I. Introduction

### 1.1 Background

The Arsenal Exchange Model (AEM) is one of the most widely used strategic force analysis models in the defense community. It is specifically designed to address strategic force analysis problems such as (Bozovich and others, 1993:4):

- strategic weapon system analysis
- strategic nuclear policy support
- arms control analysis
- force management analysis
- intelligence support
- general strategic calculations

The AEM performs allocations of weapons to targets for either side in a scenario and evaluates damage attained by the allocations (Bozovich and others, 1993:4). Because an integer number of weapons must be allocated to an integer number of targets, the problem AEM seeks to solve is an integer programming (IP) problem. However, the problem is so large, often involving more than a million general integer variables, that the time required to find an optimal integer solution would be prohibitive.

Fortunately, the model users do not require the optimal integer solution; rather, they require a good feasible integer solution quickly. In order to provide a feasible integer solution in a reasonable amount of time, AEM drops the integer requirements and solves the resulting

linear programming (LP) problem. AEM then takes the noninteger variables in the continuous LP solution and rounds them down to the nearest integer. Since this rounding process leaves weapons unused and targets uncovered, it seems likely that the method could be improved (Cotsworth, 1993b).

## *1.2 AEM Description*

This section provides a brief description of AEM. It begins by presenting the inputs that the user must provide to the model. Next, this section discusses AEM processing of these inputs and the formulation of a mathematical model. Finally, model solution and outputs are presented.

*1.2.1 User Inputs.* Two types of inputs are provided by the user: information regarding both the weapons to be allocated and targets to be covered, and information on the requirements which the user has for the allocation.

*1.2.1.1 Weapons and Targets.* The user divides the weapons into types; for example, B-52s, Minuteman IIIs, or Trident D-5s. The user then inputs to AEM the types of weapons which are available, the number of weapons of each type, and the characteristics of each weapon type. Weapon characteristics include: yield, reliability, and accuracy. The user also divides the targets into classes; for example, industrial, military, or leadership. The user then inputs the classes of targets which are available, the number of targets in each class, and the characteristics of each target class. Target characteristics

include: hardness, radius, and value (Bozovich and others, 1993:149-151).

*1.2.1.2 Requirements.* Users will normally have a list of requirements for the allocation; for example, a user may require that all targets be covered. These requirements may be added as goals which the allocation should satisfy as closely as possible or as constraints which the allocation must satisfy. Therefore, failure to satisfy a goal affects the goodness of the allocation, while failure to satisfy a constraint affects the feasibility of the allocation.

*1.2.2 Processing.* AEM takes the information input by the user and constructs the allowable combinations of weapon types and target classes. These combinations are called strategies and they are the decision variables in AEM. An allocation is completely defined by specifying the number of times each strategy is used (Bozovich and others, 1993:154-157).

For each allowable strategy AEM calculates the probability that the target will be destroyed (probability of kill, or PK). This calculation is performed by a subroutine called PDCALC, and is based upon the weapon and target characteristics provided by the user. The probability of kill is then multiplied by the value of the target to give the damage expectancy (DE); (Cotsworth and Garrett, 1991:Sec II, 2-15).

*1.2.3 Model Formulation.* The basic problem is to find a feasible allocation of weapons to targets which maximizes DE. This problem may be formulated as:

$$\max \sum_{i=1}^m DE_i x_i \quad (1.1)$$

$$s.t. \sum_{i \in ST_j} x_i \leq T_j \quad j=1, \dots, n \quad (1.2)$$

$$\sum_{i \in SW_k} B_{ki} x_i \leq W_k \quad k=1, \dots, p \quad (1.3)$$

$$x_i \geq 0 \quad i=1, \dots, m \quad (1.4)$$

where:

- $m$  = The number of allowable strategies.
- $n$  = The number of target classes.
- $p$  = The number of weapon types.
- $B_{ki}$  = The number of type  $k$  weapons used in strategy  $i$ .
- $DE_i$  = The damage expectancy resulting from one use of strategy  $i$ .
- $W_k$  = The number of type  $k$  weapons.
- $SW_k$  = The set of strategies which use weapon type  $k$ .
- $T_j$  = The number of class  $j$  targets.
- $ST_j$  = The set of strategies which attack target class  $j$ .
- $x_i$  = The number of times strategy  $i$  is used in the allocation.

Constraints (1.2) are the target constraints. They ensure that, for each target class, the number of targets attacked by the strategies is no greater than the number of targets available. Constraints (1.3) are the weapon constraints. They ensure that for each weapon type, the number of weapons used in the strategies is no greater than the number of weapons available. To this basic formulation AEM adds any constraints which were input by the user. Goals input by the user are added to the basic formulation using goal programming (see chapter 2).

*1.2.4 Solution and Output.* AEM contains a subroutine called GULP which uses a generalized upper bounding routine to solve the linear programming model (Cotsworth, 1991:58-68). Because the problem is solved as an LP, generally the solution will contain some noninteger valued variables (strategies). If the user requires an integer solution, AEM rounds the noninteger variables in the continuous solution down to the nearest integer. While the integer solution produced by this method will not violate the weapon or target constraints, it leaves weapons unused and targets uncovered, so this integer solution could likely be improved.

### *1.3 Problem Definition*

The goal of this research is to develop a better method for converting the continuous solution produced by AEM into a feasible integer solution. The method developed must produce a solution within a reasonable amount of time.

#### *1.4 Scope*

For this thesis, a better method is defined as a method which produces feasible integer solutions that are closer to the continuous solution in terms of damage expectancy, target coverage, and goal performance than the solutions produced by the current method for determining an integer solution.

Although a user may add constraints to the problem, it is assumed that all user requirements are added as goals. Thus, a solution is considered feasible if it does not violate any weapon or target constraint.

#### *1.5 Format*

Chapter II provides a brief explanation of goal programming and reviews a heuristic which has been developed to find feasible integer solutions for another integer programming problem. In Chapter III, the way AEM handles goals is described in more detail. In Chapter IV, the method developed to produce good feasible integer solutions from the continuous AEM solution is presented. Chapter V discusses the implementation and results. Chapter VI lists conclusions and recommendations.



## II. Literature Review

### 2.1 Introduction

This chapter first provides a brief discussion of goal programming, then examines a heuristic which has been developed to provide feasible integer solutions for the generalized assignment problem.

### 2.2 Goal Programming

Normally, it is not possible to completely meet or satisfy all of the goals a user has for an allocation; for example, there may not be enough weapons available to provide the desired level of coverage for each target class. To arrive at the best allocation, consistent with the competing goals, AEM uses goal programming.

In general, there are three possible outcomes for a goal; it may be met exactly, it may be exceeded by some amount  $d^+$ , or it may be underachieved by some amount  $d^-$ . Assume that it is our goal for the sum of  $x_1$ ,  $x_2$ , and  $x_3$  to equal five. We may write:

$$x_1 + x_2 + x_3 + d^- - d^+ = 5$$

$$d^-, d^+ \geq 0$$

If  $d^+ > 0$ , then  $x_1 + x_2 + x_3 > 5$  and if  $d^- > 0$ , then  $x_1 + x_2 + x_3 < 5$ . By assigning weights  $w^+$  and  $w^-$  to  $d^+$  and  $d^-$  in the objective function, the LP will try to achieve the goal ( $x_1 + x_2 + x_3 = 5$ ) as closely as possible by minimizing the weighted sum of the deviations. If the goal were  $x_1 + x_2 + x_3 \geq 5$ , then  $w^+$  would be set to zero, while if the goal were

$x_1 + x_2 + x_3 \leq 15$ , then  $w^-$  would be set to zero (Ravindran and others, 1987:198-199).

The general goal programming model can be expressed as follows:

$$\min \sum_{r=1}^q (w_r^+ d_r^+ + w_r^- d_r^-) \quad (2.1)$$

$$\text{s. t.} \quad \sum_{i=1}^n a_{ri} x_i + d_r^- - d_r^+ = b_r \quad r=1, \dots, q \quad (2.2)$$

$$x_i, d_r^-, d_r^+ \geq 0 \quad i=1, \dots, m \quad r=1, \dots, q \quad (2.3)$$

where there are  $q$  goals,  $a_{ri}$  is the coefficient associated with the  $i$ th decision variable in goal  $r$ , and  $b_r$  is the right-hand-side of goal  $r$ . Once the weights have been specified, the goal programming problem is reduced to a linear programming problem (Ravindran and others, 1987:199). Unfortunately, it may be nearly impossible for the user to determine the appropriate weights; for example, should military targets be twice as important as industrial targets, or should they be 2.5 times as important?

In preemptive goal programming, instead of assigning relative weights to the goals, the goals are assigned to priority levels. All goals at the same priority level have the same weight. The objective function for the preemptive goal program is as follows:

$$\min \sum_k (P_k \sum_i (d_i^- + d_i^+))$$

where  $P_k$  represents priority  $k$  and  $P_k \gg P_{k+1}$  (Ravindran and others, 1987:200).

In practice, the  $P_k$ 's would cause computational problems and so they are not used. Instead, preemptive goal programming methods have been developed which are based on the fact that preemptive priorities imply that higher priority goals must be optimized before lower priority goals are considered. One such method is the partitioning algorithm developed by Arthur and Ravindran (Ravindran and others, 1987:203).

In the partitioning algorithm, a series of linear programming subproblems is solved using the solution of the higher priority subproblem as the starting solution for the lower priority subproblem. The first subproblem consists of those goals assigned to the highest priority and the corresponding terms in the objective function. This subproblem is solved and the optimal tableau is examined for alternate optimal solutions. If the subproblem has no alternate optimal solutions, then the solution to the subproblem is the solution to the original problem. If alternate optimal solutions do exist, then all nonbasic columns which have a positive relative cost are removed from the tableau. This is done because a nonbasic variable with a positive relative cost in the optimal tableau cannot enter the basis to form an alternate optimal solution. The next subproblem is then constructed by adding the next highest priority goals to the tableau and adding the corresponding terms to the objective function. The algorithm continues in this way until no alternate optimal solutions exist for a subproblem, or until all the goals have been considered (Ravindran and others, 1987:203-204).

### 2.3 The Generalized Assignment Problem

The generalized assignment problem (GAP) involves  $m$  machines and  $n$  jobs. A machine may be assigned to more than one job, but a job must be assigned to exactly one machine. The objective is to find the assignment of jobs to machines that minimizes cost.

The problem is formulated as follows:

$$\min \sum_{i=1}^m \sum_{j=1}^n c_{ij} x_{ij} \quad (2.4)$$

$$\text{s.t.} \quad \sum_{j=1}^n a_{ij} x_{ij} \leq b_i \quad i = 1, \dots, m \quad (2.5)$$

$$\sum_{i=1}^m x_{ij} = 1 \quad j = 1, \dots, n \quad (2.6)$$

$$x_{ij} \in \{0, 1\} \quad i = 1, \dots, m \quad j = 1, \dots, n \quad (2.7)$$

where  $c_{ij}$  is the cost of performing job  $j$  on machine  $i$ ,  $a_{ij}$  is the machine capacity expended by performing job  $j$  on machine  $i$ , and  $b_i$  is the capacity of machine  $i$ . The variable  $x_{ij}$  equals 1 if machine  $i$  is used to perform job  $j$ , otherwise  $x_{ij}$  equals 0 (Cattrysse and Wassenhove, 1992:261).

The linear programming relaxation of the GAP (LGAP) is obtained by replacing constraint (2.7) with  $x_{ij} \geq 0$ . In general, the solution of

the LGAP will contain some jobs which are split between several machines. Benders and van Nunen show that the number of non-unique assignments is less than or equal to the number of machines which are used to capacity. This suggests that the LGAP solution may be a good starting point for a heuristic (Benders and van Nunen, 1983:48).

One heuristic which builds upon the LGAP solution is Trick's LR-Heuristic. In this heuristic, the unsplit jobs are fixed, leaving only the split jobs to be scheduled. Variables  $x_{ij}$  are deleted from the problem if the remaining unused capacity of machine  $i$  is not sufficient to perform job  $j$ . The resulting LGAP is solved and, if necessary, the procedure is repeated. Trick shows that the procedure will have to be repeated at most  $m$  times (Trick, 1992:140).

## 2.4 Conclusion

Since the problem AEM is designed to solve is different from the GAP, Trick's LR-Heuristic will not be directly applicable to AEM solutions. However, the method developed in this thesis is similar to Trick's in that it begins with the solution of the LP relaxation, the integer portion of the solution is fixed, and a new problem is formulated to reassign the resources associated with the noninteger portions of the continuous solution.

### *III. Goal Programming in AEM*

#### *3.1 Introduction*

This chapter covers aspects of goal programming that are unique to AEM. First, the types of goals used in AEM are discussed along with the method AEM uses to prevent duplicate rows in the constraint matrix. Then, the method AEM uses to accomplish preemptive goal programming is discussed.

#### *3.2 Goals*

In addition to maximizing DE when assigning weapons to targets, the user may have other goals for the allocation; for example, achieving at least a 0.8 average damage expectancy against industrial targets. In AEM, these goals are called hedges. There are four types of hedges (Bozovich and others, 1993:138):

1. Requirements pertaining to the total damage achieved on a specified set of target classes by a specified set of weapon types. An example of this is, using weapon types 2 and 3, achieve an average damage expectancy of at least 0.8 against class A targets.
2. Requirements pertaining to the total number of allocatable weapon reentry vehicles, from a specified set of weapon types, allocated against a specified set of target classes. An example of this is, use no more than one-half of the available type 2 weapons against class A targets.

3. Requirements pertaining to the total number of targets hit by a specified set of weapon types. An example of this is, attack at least one-half of the class A targets using type 2 weapons.

4. Requirements that all strategies for the specified weapon/target combinations satisfy a set of criteria. The criteria deal with the amount of damage obtained in the strategy, the number of weapons involved, and the presence of certain weapon types in the strategy. An example of this is, attack class A targets using strategies with damage expectancies of at least 0.8 .

Users often have goals which are similar to constraints in the problem. For example, a user may wish to completely cover all class A targets. This goal may be written as:

$$\sum_{i \in S_A} x_i + d^- = T_A$$

where  $S_A$  is the set of strategies that attack target class A,  $T_A$  is the number of class A targets, and the user wishes to minimize the deviation ( $d^-$ ). The target constraint for class A targets is:

$$\sum_{i \in S_A} x_i + s = T_A$$

where  $s$  is the slack variable for this constraint. If the goal is met, then  $d^-$  and  $s$  will both be zero and AEM will remove the columns associated with these variables from the tableau. Once the variables  $d^-$  and  $s$  have been removed, the rows associated with the goal and with the constraint may both be written as:

$$\sum_{i \in S_A} x_i = T_A$$

Duplicate or redundant rows in the tableau will result in a singular constraint matrix which causes difficulties for the LP solver used in AEM. To prevent this situation, AEM multiplies the right-hand-side of all goals by 0.9995. This causes the goals and constraints to differ, but does not change the goals enough to cause a significant change in the solution. That is, the goal performance should only change by 0.05 percent (Cotsworth, 1993a).

### 3.3 Preemptive Goal Programming

AEM accomplishes preemptive goal programming by solving a series of linear programming subproblems. The first subproblem may be formulated as:

$$\max \quad \sum_{i=1}^n DE_i x_i - \sum_{h \in P_1} M(w_h^+ d_h^+ + w_h^- d_h^-)$$

$$s.t. \quad \sum_{i \in S_j} x_i \leq T_j \quad j=1, \dots, n$$

$$\sum_{i \in SW_k} B_{ki} x_i \leq W_k \quad k=1, \dots, p$$

$$\sum_{i=1}^n a_{hi} x_i + d_h^- - d_h^+ = 0.9995 b_h \quad h=1, \dots, q$$



$$x_i, d_h^-, d_h^+ \geq 0 \quad i=1, \dots, m \quad h=1, \dots, q$$

where:

- $m$  = The number of possible strategies.
- $n$  = The number of target classes.
- $p$  = The number of weapon types.
- $q$  = The number of goals for the allocation.
- $DE_i$  = The damage expectancy resulting from one use of strategy  $i$ .
- $x_i$  = The number of times strategy  $i$  is used in the allocation.
- $M$  = A very large positive number.
- $w_h^+$  = 0, if goal  $h$  is a  $\geq$  goal, and 1 otherwise.
- $w_h^-$  = 0, if goal  $h$  is a  $\leq$  goal, and 1 otherwise.
- $d_h^+$  = The positive deviation variable associated with goal  $h$ .
- $d_h^-$  = The negative deviation variable associated with goal  $h$ .
- $P_1$  = The set of goals assigned to priority one.
- $ST_j$  = The set of strategies that attack target class  $j$ .
- $T_j$  = The number of class  $j$  targets.
- $SW_k$  = The set of strategies that use weapon type  $k$ .
- $W_k$  = The number of type  $k$  weapons.
- $B_{ki}$  = The number of type  $k$  weapons expended by one use of strategy  $i$ .
- $a_{hi}$  = The coefficient associated with strategy  $i$  in goal  $h$ .
- $b_h$  = The right-hand-side of goal  $h$ .

In the optimal tableau for this subproblem, if a priority one goal has been met, the weighted deviation variables associated with the goal

have a value of zero. If a priority one goal cannot be met, a weighted deviation variable associated with the goal will be nonzero in the optimal tableau. In this case, the right-hand-side of the goal is adjusted so as to make the deviation variable have a value of zero. Once the weighted deviation variables have been set to zero, they are removed from the tableau. This prevents the solutions of subsequent subproblems from deviating from the level of goal achievement reached in the current subproblem. The weighted deviation variables associated with the priority two goals are then added to the objective function. The process is continued until all of the goals have been added. The solution to the final subproblem is the optimal allocation. The optimal allocation generally will not satisfy all of the goals, but it will come as close as possible to satisfying each of the goals in the order of their priority (Cotsworth and Garrett, 1991:Sec II, 32-33).

## *IV. Methodology*

### *4.1 Introduction*

In this chapter, the optimal integer solution is examined to see how it compares with the optimal continuous solution. Then, the current solution method is reviewed. Next, a method which improves upon the current methodology is developed. The model formulation and solution procedure for this method is then presented. Finally, a revised solution method and solution procedure are presented.

### *4.2 The Optimal Integer Solution*

For an IP which is a maximization problem, the solution to its LP relaxation provides an upper bound on the objective function value of the IP. For an IP which is a minimization problem, the LP relaxation provides a lower bound on the objective function value. (Winston, 1991:458).

Because AEM uses preemptive goal programming, the continuous solution is produced by solving a series of subproblems. Priorities are assigned to the goals and the subproblems seek to minimize the deviation from the goals in order of priority. The objective of each subproblem is to minimize the deviation from the goals at the current priority level without increasing the deviations from the higher priority goals. For example, assume the highest priority goal desires the average damage expectancy of class A targets to be at least 0.8. If it is possible to achieve this level of damage expectancy, then all subsequent subproblems are constrained to also produce an average damage expectancy of at least 0.8. If the greatest possible average damage expectancy against class A

targets is only 0.79, then all subsequent subproblems are constrained to produce an average damage expectancy of 0.79 against class A targets.

To see how LP relaxation affects the solution, again assume that the highest priority goal is to achieve an average damage expectancy of 0.8 against class A targets. If this goal cannot be met, then the integer solution will have a deviation which is at least as large as the deviation produced by the LP relaxation. For example, the optimal integer solution may achieve an average damage expectancy of 0.78, while the LP relaxation may achieve 0.79. In this case, subsequent subproblems will be less constrained in the integer solution; they only have to achieve an average damage expectancy of 0.78 while, for the continuous solution, 0.79 is required. Although the integer solution has not done as well as the continuous solution on this goal, the remaining subproblems are less constrained, and therefore, the integer solution may do better than the continuous solution on the remaining goals.

If the goal can be met, then, in the continuous solution, all subsequent subproblems are also required to achieve an average damage expectancy against class A targets of at least 0.8. But, in the integer solution, it may not be possible to achieve a damage expectancy of exactly 0.8. It may be that the lowest damage expectancy which can be achieved by an integer solution, while still achieving at least 0.8, is 0.81. This means that subsequent subproblems are more constrained for the integer solution; they must produce an average damage expectancy of at least 0.81 while, in the continuous solution, only 0.8 is required. Although the integer solution may meet this goal at a higher level than the continuous solution, the remaining subproblems are more constrained,

and therefore, the integer solution may do worse than the continuous solution on the remaining goals.

Because of the complex balancing which occurs between the goals, it is difficult to predict how the optimal integer solution will compare to the continuous solution. The integer solution may do worse than the continuous solution on some goals and better on others. Also, because the final subproblem may be more or less constrained, the integer solution may have a damage expectancy which is higher, lower, or exactly the same as in the continuous solution.

#### *4.3 Current Solution Method*

AEM currently produces feasible integer solutions by starting with the continuous solution and rounding all of the noninteger valued variables down to the closest integer. This solution is fairly good because the allocations generally involve thousands of weapons and thousands of targets so the number of weapons and targets which contribute to each goal is normally large. Also, most of the variables in the continuous solution are usually integer. Therefore, the number of weapons lost in the rounding process is so small compared to the total number of weapons being allocated that they do not make a significant difference in the goal performance.

#### *4.4 A Better Rounding Method*

Although rounding down produces a feasible integer solution, it leaves weapons unused and targets uncovered. If, instead of rounding all the noninteger variables down, some were rounded up, the damage expectancy and target coverage would be improved, as compared to the

current solution method, and the goal performance would still be about the same as in the continuous solution. In fact, since most goals require that at least a certain level of damage expectancy or target coverage be attained, and since rounding some variables up will increase both damage expectancy and target coverage, goal performance should generally be improved compared to the integer solutions obtained using the current method.

Since experience shows that the goal performance will normally be about the same as in the continuous solution, regardless of which variables are rounded up and which are rounded down, the goals may be ignored. Ignoring the goals relaxes the constraints on the problem. This means that the resulting solution may actually be better in some ways than the optimal integer solution; for example, the performance may be better for some goals or the damage expectancy may be higher. However, it is important to remember that the solution may do better on some goals only because it is allowed to do worse on others.

Since damage expectancy is an important measure of how good a solution is, it would at first seem desirable to round the variables so as to maximize damage expectancy. However, maximizing damage expectancy tends to produce a solution which attacks the soft targets and ignores the hard targets. Since the harder targets, like missile silos, are often the more important ones, simply maximizing damage expectancy may not produce the best solutions.

When the continuous solution is produced, the objective is to maximize damage expectancy. To ensure that the important target classes are covered, the user includes covering these target classes as a goal.

A similar goal programming approach can be used to ensure that the variables are rounded so as to cover the important target classes.

Since covering the important target classes was a goal of the original problem, the continuous solution will have covered them, if it was possible to do so. Therefore, an integerization method which covers the same target classes as the continuous solution will also cover the important target classes.

#### 4.5 Model Formulation

An integer goal program may be constructed that identifies which variables to round up and which to round down so as to maximize damage expectancy while completely covering those target classes which were completely covered in the continuous solution. This Rounding IP may be formulated as follows:

$$\max \quad \sum_{i=1}^m DE_i x_i - Md^- \quad (4.1)$$

$$s. t. \quad \sum_{i \in ST_j} x_i \leq T_j \quad j=1, \dots, n \quad (4.2)$$

$$\sum_{i \in SW_k} B_{ki} x_i \leq W_k \quad k=1, \dots, p \quad (4.3)$$

$$\sum_{i \in SC} x_i + d^- = \sum_{j \in TC} T_j \quad (4.4)$$

$$X_i \in (0,1) \quad i=1, \dots, m \quad (4.5)$$

$$d^- \geq 0 \quad (4.6)$$

where:

- $m$  = Number of strategies with noninteger value in the continuous solution.
- $n$  = Number of target types which were attacked using noninteger valued strategies in the continuous solution.
- $p$  = Number of weapon types which were allocated using a noninteger valued strategy in the continuous solution.
- $M$  = A large positive number.
- $d^-$  = The deviation variable associated with the goal (4.4).
- $B_{ki}$  = Number of type  $k$  weapons used in strategy  $i$ .
- $DE_i$  = Damage expectancy resulting from one use of strategy  $i$ .
- $S$  = Set of strategies with noninteger values in the continuous solution.
- $SC$  = Subset of  $S$  which attack target classes that were completely covered in the continuous solution.
- $ST_j$  = Subset of  $S$  that attack target class  $j$ .
- $SW_k$  = Subset of  $S$  that use weapon type  $k$ .
- $T_j$  = Number of targets of class  $j$  left to be attacked.
- $TC$  = Set of target classes that were completely covered in the continuous solution.



$W_k$  = Number of weapons of type  $k$  left to be allocated.

$X_i$  = 0, if strategy  $i$ 's value should be rounded down.

1, if strategy  $i$ 's value should be rounded up.

The constraints (4.2) are the target constraints. The target constraints ensure that, for each target class, the number of targets attacked is no greater than the number of targets available.

Constraints (4.3) are the weapon constraints. The weapon constraints ensure that, for each weapon type, the number of weapons allocated is no greater than the number of weapons available. Constraint (4.4) is the goal of completely covering those target classes which were completely covered in the continuous solution. The objective function (4.1) seeks to maximize the total damage expectancy minus a penalty which is assessed for deviating from the goal.

The IP constructed will have  $n+p+1$  constraints and  $m$  binary variables. Since  $m$  will generally be less than 50, and  $n$  and  $p$  will each generally be less than 25, this problem can be solved in a reasonable period of time using an integer solver.

#### 4.6 Solution Procedure

This section presents the solution procedure which uses the Rounding IP from section 4.5. Figure 1 shows the flow of the solution procedure.

STEP 1: Obtain the continuous solution from AEM. If all variables are integer, stop.

STEP 2: Create a *truncated* solution by rounding all noninteger variables down to the nearest integer. Set aside this truncated solution.

STEP 3: Gather information from the continuous solution.

- a. Create a new target list from those targets which were attacked in the continuous solution but are not attacked in the truncated solution. If there is a noninteger number of targets of some class, round up to the next highest integer.
- b. Create a new weapons list from those weapons which were used in the continuous solution but are not used in the truncated solution.
- c. Identify strategies with noninteger value in the continuous solution.
- d. Identify target classes which were completely covered using noninteger valued strategies.

STEP 4: Formulate the Rounding IP using information collected at STEP 3.

STEP 5: Solve the Rounding IP.

STEP 6: Combine solution obtained in STEP 5 with truncated solution from STEP 2.

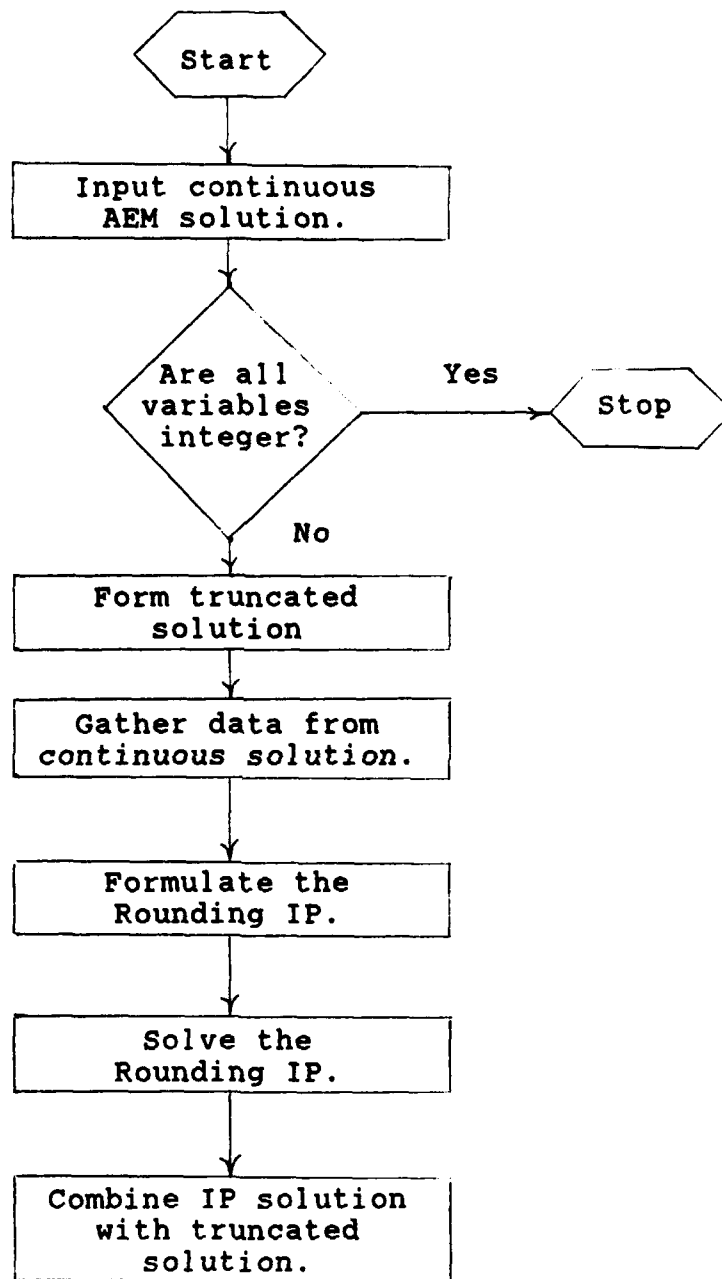


Figure 1. Flow Diagram, Solution Procedure

#### 4.7 Revised Solution Method

Experimenting on small problems, it was discovered that the LP relaxation of the Rounding IP, obtained by replacing the constraints (4.5) with:

$$x_i \geq 0 \quad i = 1, \dots, m$$

frequently produces an integer solution. Since AEM contains a linear program solver, but does not contain an integer program solver, a solution method was developed using the LP relaxation of the Rounding IP.

Although the LP relaxation of the Rounding IP should frequently produce an integer solution, because AEM modifies all goals by multiplying the right-hand-side by 0.995, and due to numerical precision problems, the solution obtained using AEM will not be exactly integer. To produce an intermediate integer solution, the LP relaxation of the Rounding IP is solved and the noninteger valued variables are rounded to the nearest integer.

#### 4.8 Revised Solution Procedure

This section presents a solution procedure which uses the LP relaxation of the Rounding IP. Figure 2 shows the flow of the revised solution procedure.

STEP 1: Obtain the continuous solution from AEM. If all variables are integer, stop.

STEP 2: Create a truncated solution by rounding all noninteger variables down to the nearest integer. Set aside this truncated solution.

STEP 3: Gather information from the continuous solution.

- a. Create a new target list from those targets which were attacked in the continuous solution but are not attacked in the truncated solution. If there is a noninteger number of targets of some class, round up to the next highest integer.
- b. Create a new weapons list from those weapons which were used in the continuous solution but are not used in the truncated solution.
- c. Identify strategies with noninteger value in the continuous solution.
- d. Identify target classes which were completely covered using noninteger valued strategies.

STEP 4: Formulate the LP relaxation of the Rounding IP using information collected at STEP 3.

STEP 5: Solve the LP relaxation of the Rounding IP.

STEP 6: Produce an intermediate integer solution by rounding all noninteger valued variables to the nearest integer.

STEP 7: Combine the intermediate integer solution obtained in STEP 6 with the truncated solution obtained in STEP 2.

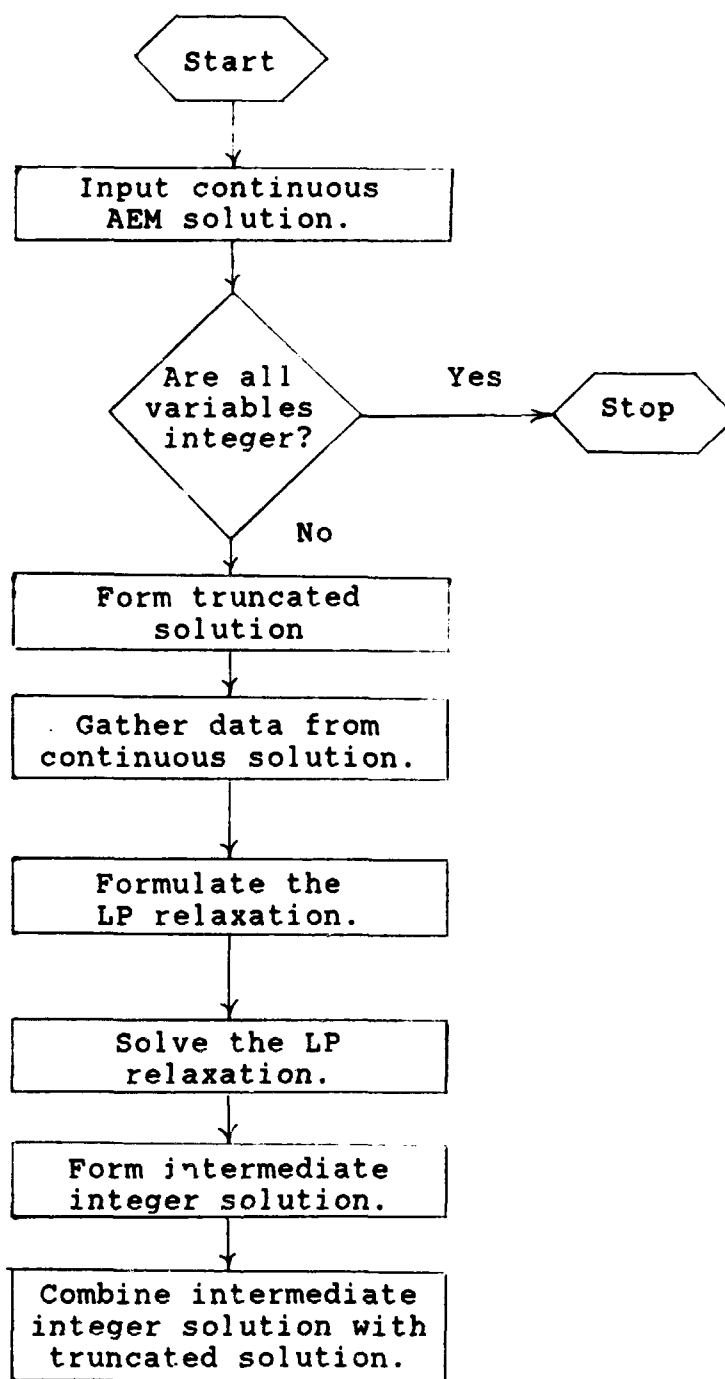


Figure 2. Flow Diagram, Revised Solution Procedure



## *V. Implementation and Testing*

### *5.1 Implementation*

The revised solution method presented in section 4.7 was implemented in AEM by Mr. William L. Cotsworth, President of AEM Services Incorporated, under the sponsorship of the Air Force Studies and Analyses Agency (USAFSAA).

### *5.2 Testing*

The method was tested on three test cases provided by Mr. William L. Cotsworth and one actual case provided by USAFSAA. For each case, three solutions were produced. Solution one is the continuous AEM solution. Solution two is the integer solution obtained using the current integer solution method in which all noninteger variables in the continuous solution are rounded down to the nearest integer. Solution three was obtained using the revised solution method presented in section 4.7.

*5.2.1 The First Test Case.* The first case has 8691 weapons divided into 34 weapon types and 5774 targets divided into 122 target classes. There are 33 goals and strategies may use either one or two weapons. The continuous solution contained 160 strategies, 34 of which were noninteger in value.

The goal performance for the first case is summarized in Table 1. It can be seen that, for each goal, the absolute difference in goal performance between solution 1 and solution 2 is always at least as large as the absolute difference in goal performance between solution 1

and solution 3. The largest absolute difference in goal performance between solution 1 and solution 2 is 50 percent, while the largest absolute difference in goal performance between solution 1 and solution 3 is only 0.51 percent. The average absolute difference in goal performance between solution 1 and solution 2 is 3.78 percent, while the average absolute difference in goal performance between solution 1 and solution 3 is only 0.03 percent.

Table 2 lists the damage expectancy and target coverage achieved by the three solution methods. It can be seen that, in this case, solution 1 and solution 3 have approximately the same damage expectancy and target coverage, while solution 2 leaves 22 targets uncovered and has a slightly lower damage expectancy than the other two solutions.

Table 1. Goal Performance, Test Case 1.

Goal	Solution 1	Solution 2	Solution 3	2/1 x 100	3/1 x 100
1	100.00	99.22	99.77	99.22	99.77
2	100.00	96.60	100.05	96.60	100.05
3	100.00	98.20	100.05	98.20	100.05
4	100.00	99.43	99.98	99.43	99.98
5	86.67	86.25	86.65	99.52	99.98
6	100.00	99.72	100.00	99.72	100.00
7	99.89	99.56	99.89	99.67	100.00
8	100.00	99.58	100.00	99.58	100.00
9	22.00	22.00	22.00	100.00	100.00
10	360.00	356.00	360.00	98.89	100.00
11	58.00	56.00	58.00	96.55	100.00
12	91.00	91.00	91.00	100.00	100.00

Table 1. (Continued)

Goal	Solution 1	Solution 2	Solution 3	2/1 x 100	3/1 x 100
13	18.00	18.00	18.00	100.00	100.00
14	103.00	101.00	103.00	98.06	100.00
15	726.00	720.00	726.00	99.17	100.00
16	196.00	194.00	196.00	98.98	100.00
17	4.00	3.00	4.00	75.00	100.00
18	2.00	1.00	2.00	50.00	100.00
19	333.08	331.00	333.00	99.38	99.98
20	5.00	4.00	5.00	80.00	100.00
21	91.54	89.00	92.00	97.23	100.51
22	75.29	74.99	75.29	99.60	100.00
23	80.00	78.21	80.05	97.76	100.06
24	70.00	69.86	70.00	99.80	100.00
25	60.00	59.72	59.99	99.53	99.98
26	75.29	74.99	75.29	99.60	100.00
27	80.00	78.21	80.05	97.76	100.06
28	70.00	69.86	70.00	99.80	100.00
29	60.00	59.72	59.99	99.53	99.98
30	75.29	74.99	75.29	99.60	100.00
31	80.00	78.21	80.05	97.76	100.06
32	70.00	69.86	70.00	99.80	100.00
33	60.00	59.72	59.99	99.53	99.98

Table 2. Damage Expectancy and Target Coverage,  
Test Case 1.

	Solution 1	Solution 2	Solution 3
Damage Expectancy	72.59	72.25	72.59
Target Coverage	5503.39	5481.00	5503.00

5.2.2 *The Second Test Case.* The second case has 13,432 weapons divided into 19 weapon types and 10,938 targets divided into 11 target classes. There are 39 goals and only single weapon strategies are allowed. The continuous solution contained 98 strategies, 48 of which were noninteger in value.

The goal performance for test case 2 is summarized in Table 3. It can be seen that, for each goal, the absolute difference in goal performance between solution 1 and solution 2 is always at least as large as the absolute difference in goal performance between solution 1 and solution 3. The largest absolute difference in goal performance between solution 1 and solution 2 was 3.75 percent, while the largest absolute difference in goal performance between solution 1 and solution 3 is only 1.54 percent. The average absolute difference in goal performance between solution 1 and solution 2 is 0.53 percent, while the average absolute difference in goal performance between solution 1 and solution 3 is 0.1 percent.

Table 4 lists the damage expectancy and target coverage achieved by the three solution methods. It can be seen that the damage expectancy achieved by solution 3 is approximately the same as the damage expectancy achieved by solution 1, while the damage expectancy achieved by solution 2 is slightly lower. Solution 1 attacked two more targets than solution 3 and 27 more targets than solution 2.

Table 3. Goal Performance, Test Case 2

Goal	Solution 1	Solution 2	Solution 3	2/1 x 100	3/1 x 100
1	100.05	99.97	100.05	99.92	100.00
2	77.90	77.79	77.89	99.86	99.99
3	100.00	99.66	99.99	99.66	99.99
4	51.19	50.94	51.16	99.52	99.96
5	69.97	69.87	69.96	99.86	99.99
6	70.00	69.51	70.13	99.30	100.19
7	0.00	0.00	0.00	100.00	100.00
8	79.42	79.10	79.42	99.60	100.00
9	50.00	49.97	49.97	99.94	99.94
10	0.00	0.00	0.00	100.00	100.00
11	72.10	70.92	72.36	98.37	100.36
12	0.00	0.00	0.00	100.00	100.00
13	74.99	72.18	75.01	96.25	100.03
14	73.24	73.10	73.24	99.80	100.00
15	62.82	62.23	62.82	99.07	100.00
16	48.36	47.94	48.47	99.13	100.21
17	49.96	49.19	49.19	98.46	98.46
18	58.97	57.35	58.87	97.26	99.83
19	49.99	49.71	50.05	99.43	100.11
20	71.69	71.28	71.68	99.43	99.99
21	58.48	58.12	58.45	99.40	99.96
22	70.00	69.75	70.05	99.65	100.07
23	50.00	49.96	49.96	99.93	99.93
24	0.00	0.00	0.00	100.00	100.00
25	0.00	0.00	0.00	100.00	100.00
26	68.38	68.12	68.32	99.62	99.91
27	50.00	49.93	49.93	99.87	99.87
28	35.55	35.39	35.39	99.57	99.57

Table 3. (Continued)

Goal	Solution 1	Solution 2	Solution 3	2/1 x 100	3/1 x 100
30	0.00	0.00	0.00	100.00	100.00
31	0.00	0.00	0.00	100.00	100.00
31	0.00	0.00	0.00	100.00	100.00
32	67.57	67.53	67.57	99.94	100.00
33	71.62	71.46	71.62	99.78	99.99
34	58.56	57.48	58.66	98.16	100.16
35	51.35	51.23	51.35	99.75	100.00
36	46.17	45.94	46.15	99.51	99.95
37	199.90	199.00	200.00	99.55	100.05
38	38.08	37.92	38.07	99.58	99.97
39	0.00	0.00	0.00	100.00	100.00

Table 4. Damage Expectancy and Target Coverage,  
Test Case 2.

	Solution 1	Solution 2	Solution 3
Damage Expectancy	46.46	46.30	46.46
Target Coverage	7225.00	7198.00	7223.00

5.2.3 *The Third Test Case.* The third case has 29,381 weapons divided into 50 weapon types and 34,812 targets divided into 999 target classes. There are 11 goals and only single weapon strategies are allowed. The continuous solution contained 1051 strategies, 46 of which were noninteger in value.

The goal performance for test case 3 is summarized in Table 5. It can be seen that, for each goal, the absolute difference in goal performance between solution 1 and solution 2 is always at least as

large as the absolute difference in goal performance between solution 1 and solution 3. The largest absolute difference in goal performance between solution 1 and solution 2 is 0.43 percent, while the largest absolute difference in goal performance between solution 1 and solution 3 is 0.04 percent. The average absolute difference in goal performance between solution 1 and solution 2 is 0.11 percent, while the average absolute difference in goal performance between solution 1 and solution 3 is 0.01 percent.

Table 6 lists the damage expectancy and target coverage for the three solution methods. Solution 3 has approximately the same damage expectancy as solution 1, while the damage expectancy for solution 2 is slightly lower. Solution 1 attacks one more target than solution 3 and 16 more targets than solution 2.

Table 5. Goal Performance, Test Case 3.

Goal	Solution 1	Solution 2	Solution 3	2/1 x 100	3/1 x 100
1	70.00	69.94	70.00	99.92	100.00
2	80.00	79.92	80.00	99.90	100.00
3	70.00	69.93	70.00	99.90	100.00
4	60.00	59.99	59.99	99.99	99.99
5	50.00	49.99	50.00	99.98	100.01
6	40.00	39.95	40.00	99.88	99.99
7	30.00	29.91	29.99	99.71	99.97
8	20.00	19.91	20.01	99.57	100.04
9	30.00	30.00	30.00	99.99	99.99
10	40.00	40.00	40.00	99.99	99.99
11	33.94	33.93	33.94	99.98	100.00

Table 6. Damage Expectancy and Target Coverage,  
Test Case 3.

	Solution 1	Solution 2	Solution 3
Damage Expectancy	49.18	49.13	49.18
Target Coverage	19414.23	19398.00	19413.00

5.2.4 *The Actual Case.* The actual case was run at USAFSAA.

Since the case is classified, none of the specifics can be given here.

Table 7 summarizes the performance of solutions 2 and 3 as a percentage of the performance of the continuous solution. It can be seen that solution 3 approximates the continuous solution more closely than solution 2 in terms of goal performance, damage expectancy, and target coverage.

Table 7. Performance Summary for Actual Case.

	2/1 x 100	3/1 x 100
Average Goal Performance	96.18	100.02
Damage Expectancy	99.26	100.01
Target Coverage	99.17	100.01



## VI Conclusions and Recommendations

### 6.1 Conclusions

In the four cases tested, the new integer solution method proved to be an improvement over the current method in terms of damage expectancy, target coverage, and goal performance.

### 6.2 Recommendations

Because the solution method was implemented using the LP relaxation of the Rounding IP developed in section 4.5, the resulting solution is not guaranteed to be integer. In fact, due to the way AEM solves the LP, the solutions will generally not be integer. However, in the cases tested, all the variables were close to integer and, when the noninteger variables were rounded to the nearest integer, the resulting allocation was feasible. If the variables are not all close to integer, it is possible that a weapon or target could be *created* in the rounding process, and the resulting solution would be infeasible. For example, if two strategies attacking the same target were both selected 0.5 times, and then both rounded up so that they were each selected once, the resulting allocation would attack one more target than was available.

Adding an integer solver to AEM would avoid this problem; however, the risk of producing an infeasible solution may not be great enough to justify the added expense. More research needs to be done to determine under what circumstances an infeasible solution may be produced and the likelihood of these circumstances occurring.

Until this problem is solved, if the solution of the LP relaxation is not close to integer, the user should check carefully to ensure that the allocation produced by our rounding approach is feasible.

## *Bibliography*

- Benders, J. F. and J. A. E. E. van Nunen. "A Property of Assignment Type Mixed Integer Linear Programming Problems," Operations Research Letters, 2: 47-52 (June 1983).
- Bozovich, J. F. and others. The Arsenal Exchange Model (AEM) User's Manual. Englewood, Colorado: AEM Services, 1993.
- Cattrysse, Dirk G. and Luk N. Van Wassenhove. "A Survey of Algorithms for the Generalized Assignment Problem," European Journal of Operational Research, 60: 260-272 (August 1992).
- Cotsworth, William L. The Arsenal Exchange Model Mathematical Documentation. Englewood, Colorado: AEM Services, 1991.
- Cotsworth, William L. President, AEM Services Corporation, Englewood CO. Telephone Interview. 29 September 1993.
- Cotsworth, William L. President, AEM Services Corporation, Englewood CO. Personal Interview. 20 October 1993.
- Cotsworth, William L. and S. D. Garrett. The Arsenal Exchange Model Advanced Course. Englewood, Colorado: AEM Services, 1991.
- Fisher, Marshall L. and Ramchandran Jaikumar. "A Generalized Assignment Heuristic for Vehicle Routing," Networks, 11: 109-124 (November 1981).
- Ravindran, A. and others. Operations Research: Principles and Practice. New York: John Wiley and Sons, 1987.
- Trick, Michael A. "A Linear Relaxation Heuristic for the Generalized Assignment Problem," Naval Research Logistics, 39: 137-151 (March 1992).
- Winston, Wayne L. Introduction to Mathematical Programming: Applications and Algorithms. Boston: PWS Kent Publishing Company, 1991.

## Vita

Captain Daniel J. Green was born on 28 April 1964 in Seattle, Washington. He graduated from Wenatchee High School in Wenatchee, Washington in 1982 and attended Texas A&M University in College Station, Texas, graduating with a Bachelor of Science in Geophysics in May 1986. He attended the Air Force Officer Training School at Lackland AFB, Texas, receiving a reserve commission in the United States Air Force in February 1987. He served his first tour of duty at Whiteman AFB, Missouri. While assigned to Whiteman AFB, he served as a Deputy Missile Combat Crew Commander, Standardization/Evaluation Deputy Missile Combat Crew Commander, Missile Combat Crew Commander, Flight Commander, Instructor Missile Combat Crew Commander, and ICBM Officer Code Controller. He entered the School of Engineering, Air Force Institute of Technology, in August 1992.

Permanent address: 5775 Gross Drive  
Dayton, Ohio 45431

# REPORT DOCUMENTATION PAGE

Form Approved  
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE March 1994	3. REPORT TYPE AND DATES COVERED Master's Thesis
4. TITLE AND SUBTITLE AN INTEGER SOLUTION HEURISTIC FOR THE ARSENAL EXCHANGE MODEL (AEM)		5. FUNDING NUMBERS
6. AUTHOR(S) Daniel J. Green, Captain, U.S. Air Force		
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Institute of Technology WPAFB, OH 45433-6583		8. PERFORMING ORGANIZATION REPORT NUMBER AFIT/GST/ENS/94M-04
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Studies and Analyses Agency (USAFSAA) 1570 Air Force Pentagon Washington D.C. 20330-1570		10. SPONSORING / MONITORING AGENCY REPORT NUMBER

## 11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT  
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

12b. DISTRIBUTION CODE

## 13. ABSTRACT (Maximum 200 words)

The Air Force Studies and Analyses Agency sought a method for converting the continuous solution produced by the Arsenal Exchange Model into a feasible integer solution. The current integerization method leaves weapons unused and targets uncovered and could be improved. In the method developed, the noninteger valued variables in the continuous solution are rounded down to the nearest integer to produce a truncated solution. An integer goal program is then used to reallocate the weapons and targets made available in the rounding process. The truncated solution is then combined with the solution of the integer goal program to produce a feasible integer solution for the original problem. The revised solution method, using the LP relaxation of the integer goal program, was implemented in AEM. The implementation was used to solve four test cases. In all four cases, the revised solution method produced solutions that were closer to the continuous solution in terms of damage expectancy, target coverage, and goal performance than the solutions produced by the current method.

## 14. SUBJECT TERMS

Allocations, Mathematical Programming, Goal Programming, Integer Programming

## 15. NUMBER OF PAGES

51

## 16. PRICE CODE

## 17. SECURITY CLASSIFICATION OF REPORT

Unclassified

## 18. SECURITY CLASSIFICATION OF THIS PAGE

Unclassified

## 19. SECURITY CLASSIFICATION OF ABSTRACT

Unclassified

## 20. LIMITATION OF ABSTRACT

UL